Mission-Level Autonomy

Autonomous Rotorcraft Project

Michael Freed

NASA Ames Research Center

People

Apex Dev Team

Michael Freed

Michael Dalal

Will Fitzgerald

Dawn Fitzpatrick

Robert Harris

Users

NASA (several groups)
FAA
MITRE
Microanalysis and Design
Smart Information Flow Technologies
Applied Physics Lab
Dynamic Research Inc.
BBN
CMU
Stanford
George Mason University
University of Maryland
RPI

Background

Autonomy research at NASA

- Decades of investment by wide range of programs
- Particular emphasis at Ames, Intelligent Systems Division

Apex Project – reusable autonomy software

- Began in 1997, supported out of several programs

Autonomous Rotorcraft Project

- Began in 2001, as part of NASA Intelligent Systems program. Rotorcraft seen as important platform for terrestrial applications (Earth Science) and as analogue for planetary exploration vehicles.
- Early objective was to develop autonomy capabilities useful to both Army and NASA

Outline

- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Visualizing autonomy logic

Outline

- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Visualizing autonomy logic

Apex Project Background

Objectives and Approach

Objective: Reusable for diverse missions/platforms

- Capabilities enhanced by use in diverse applications by dev team and external users (~200)
- Reduced cost/difficulty by continuous improvement in software qualities, documentation, tools…
- Usability continuously improved in response to user feedback: behavior specification, visualization tools, APIs, …

Apex Project Background

Applications

Real Robot

Autonomous Rotorcraft Project

intelligent surveillance and reconnaissance

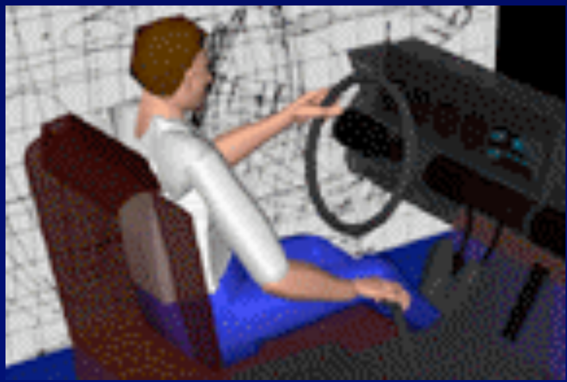
Simulated Robot

Mission Simulation Facility / REF

Riptide High-fidelity flight simulation

AuRA Wildfire detection, Earth Science

X-Plane Flight failure detection/recovery



Real Human

Astronaut Procedure Guidance

Simulated Human

CPM-GOMS HCI Analysis

VAMS Virtual Participants in HIL Simulations

MIDAS HCI Analysis

Dynamic Research Inc. Accident Analysis

Apex Project Background

System Overview

System elements

- Agent architecture, reasoning and control services, behavior representation language (PDL)
- Sherpa (autonomy logic and behavior visualization)
- Simulation engine (prototyping support)
- APIs, interop support (HLA, DOMS, UDP, TCP, XML)
- Support for install, update, portability
- Manual, sample apps, web site
- Publications

Three Layer Architecture

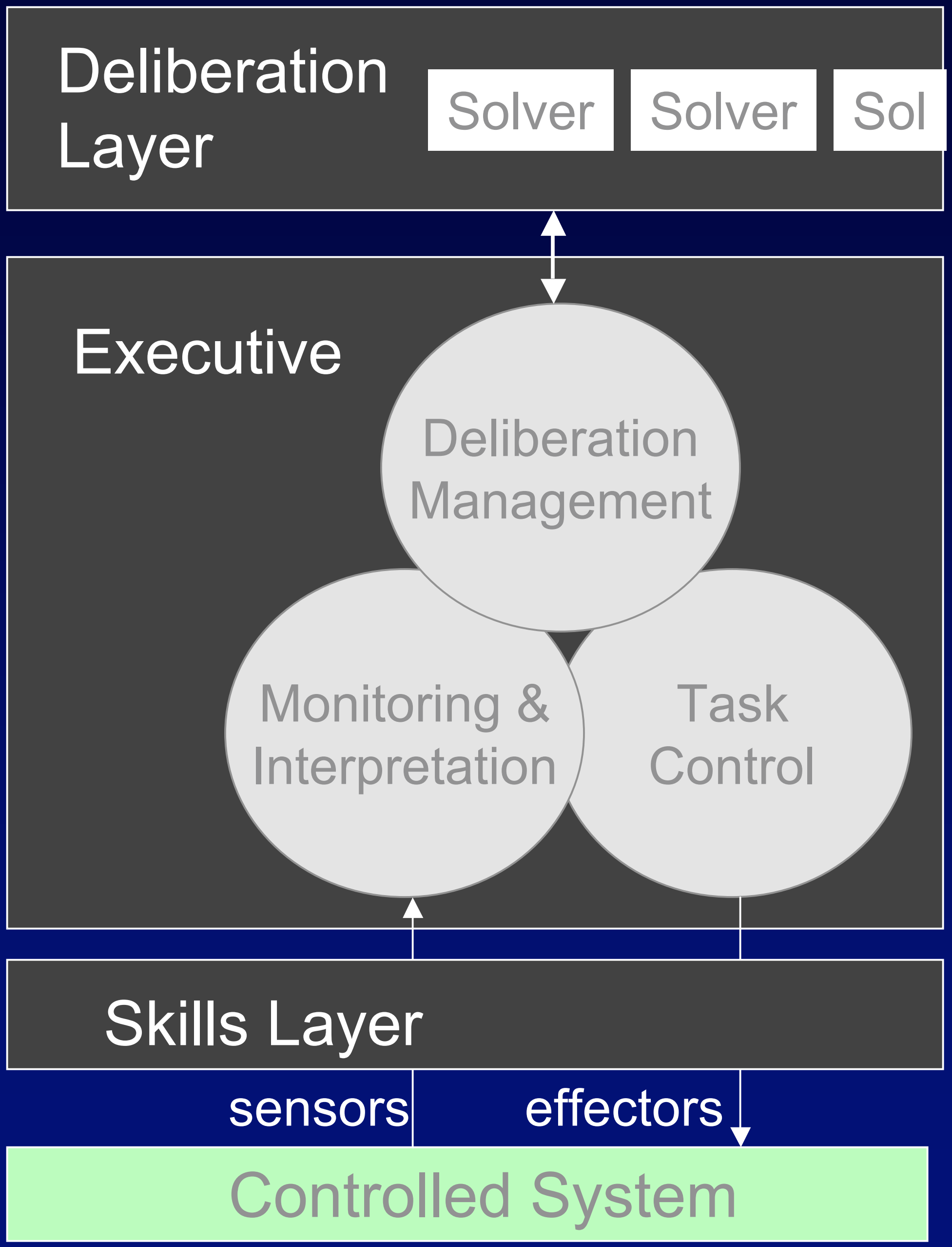
Top: slow/costly *solver* alg's
e.g. AI planner, path planner

Mid: responsive reasoning & control functions

Bottom: sensor processing & effector control

Principal design concepts

1. Separation of fast/slow
2. Separation of reusable from application-specific

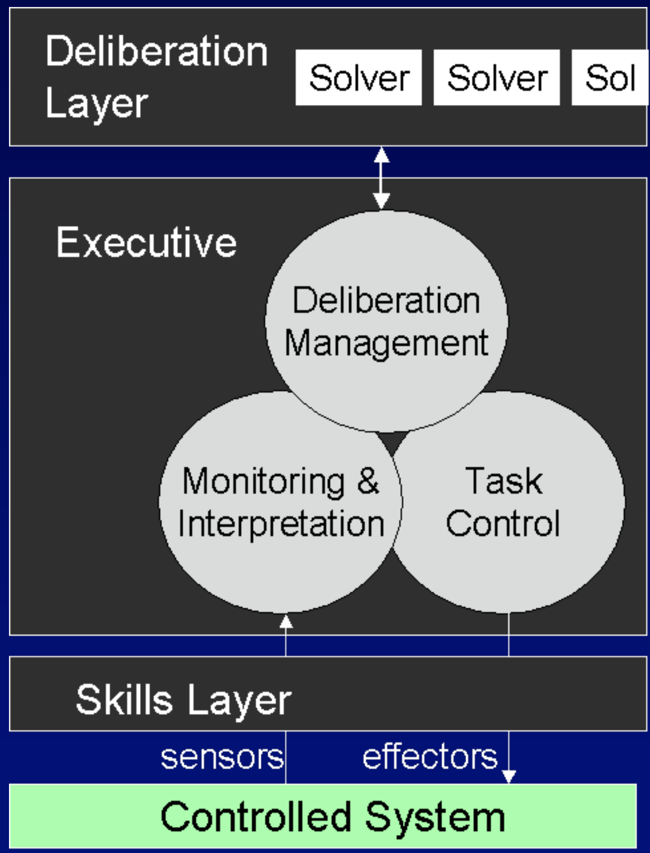


Autonomy Architecture
E.g. Autonomous Rotorcraft Project

Deliberation Layer
Periodic surveillance planning

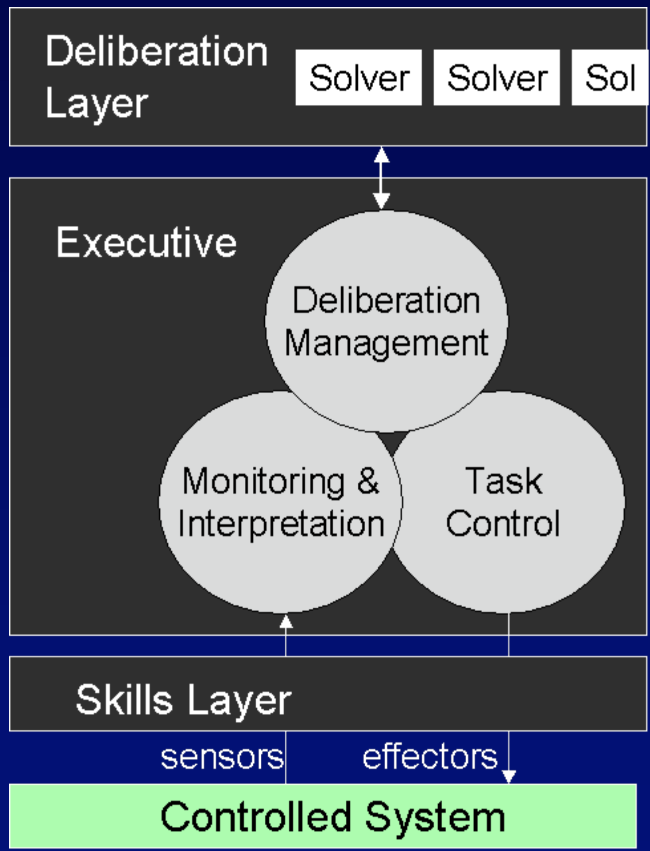
Goal Executive Layer
Basic plan execution
Tactical observation maneuvers
Monitoring and anomaly-handling
Human interaction management

Skills (application-specific)
Autopilot
Payload controllers



Reasoning and Control Services

Dispatch
Signal and process handling
Condition detection
Memory management
Refinement
Transformation
Projection
Self-callibration
Deliberation control



Outline

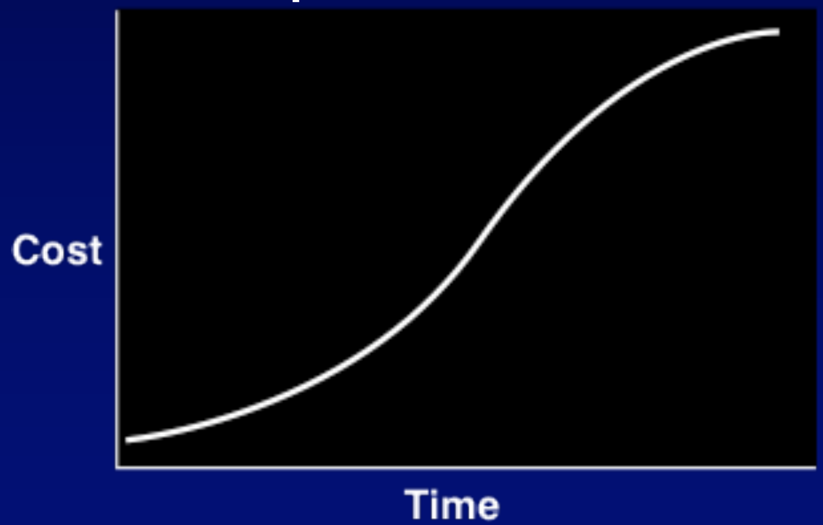
- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Visualizing autonomy logic

Mission-level Autonomy Focus

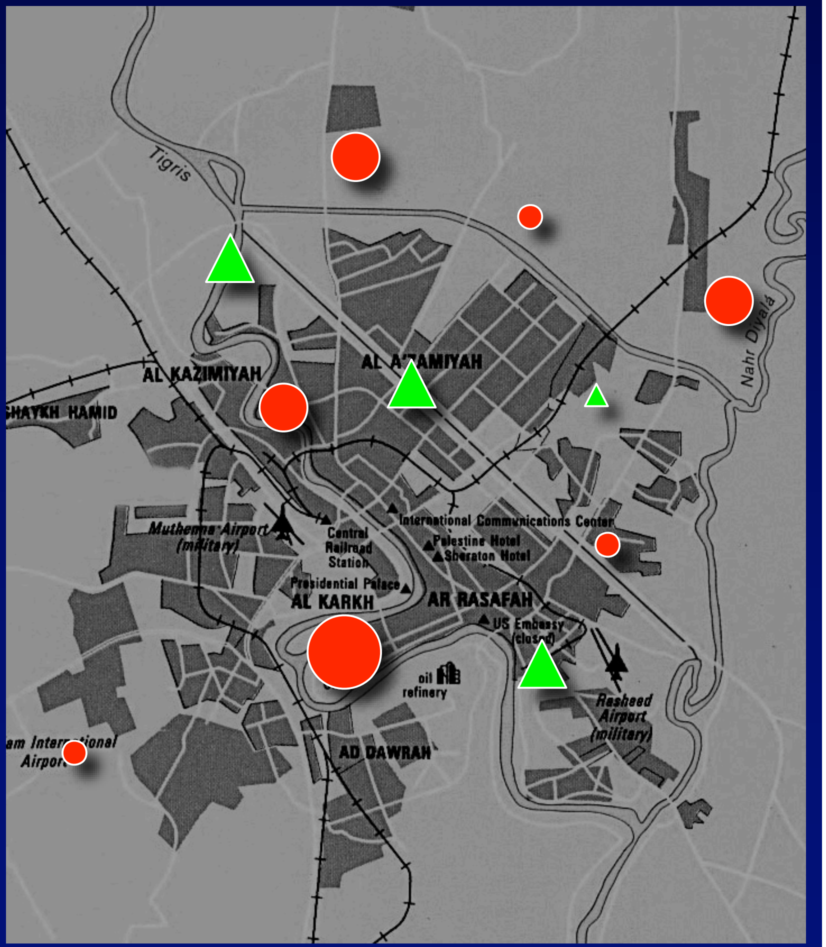
Periodic Surveillance

- Given: many targets to monitor with one/few aircraft
- Objective: early change awareness
- Decide: where to go next

Example: fire detection



Observe frequently to minimize fire detection latency

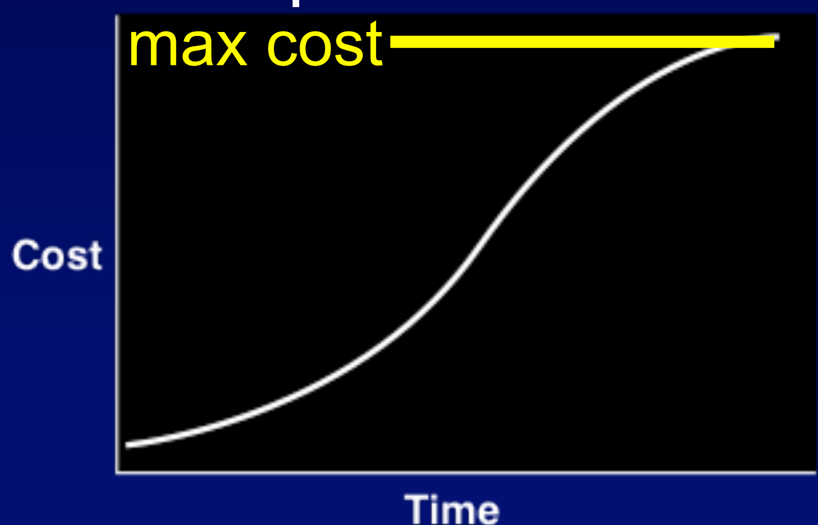


Mission-level Autonomy Focus

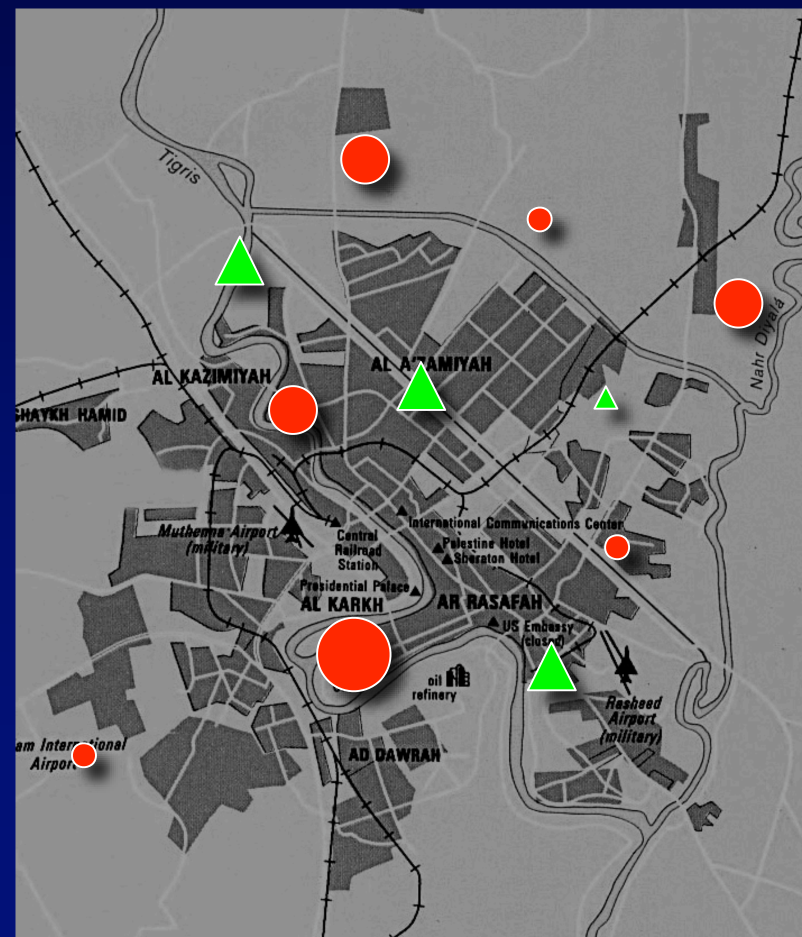
Periodic Surveillance

- Given: many targets to monitor with one/few aircraft
- Objective: early change awareness
- Decide: where to go next

Example: fire detection



Possibly visit more valuable targets more often

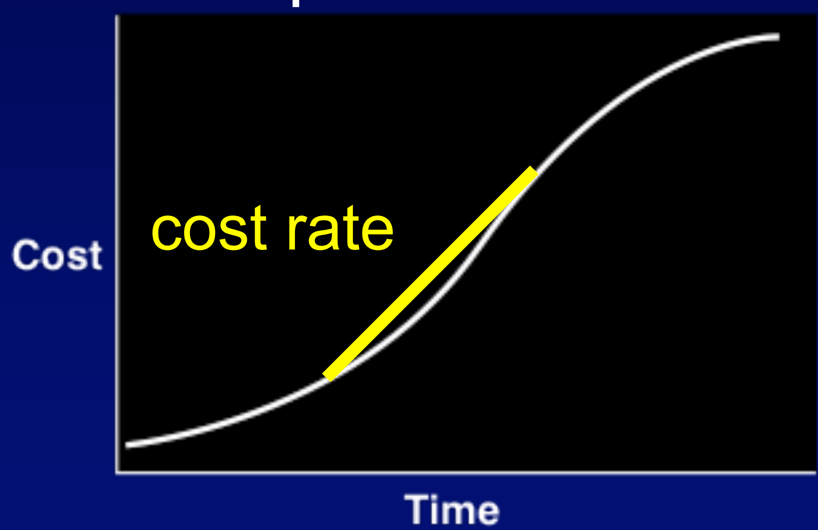


Mission-level Autonomy Focus

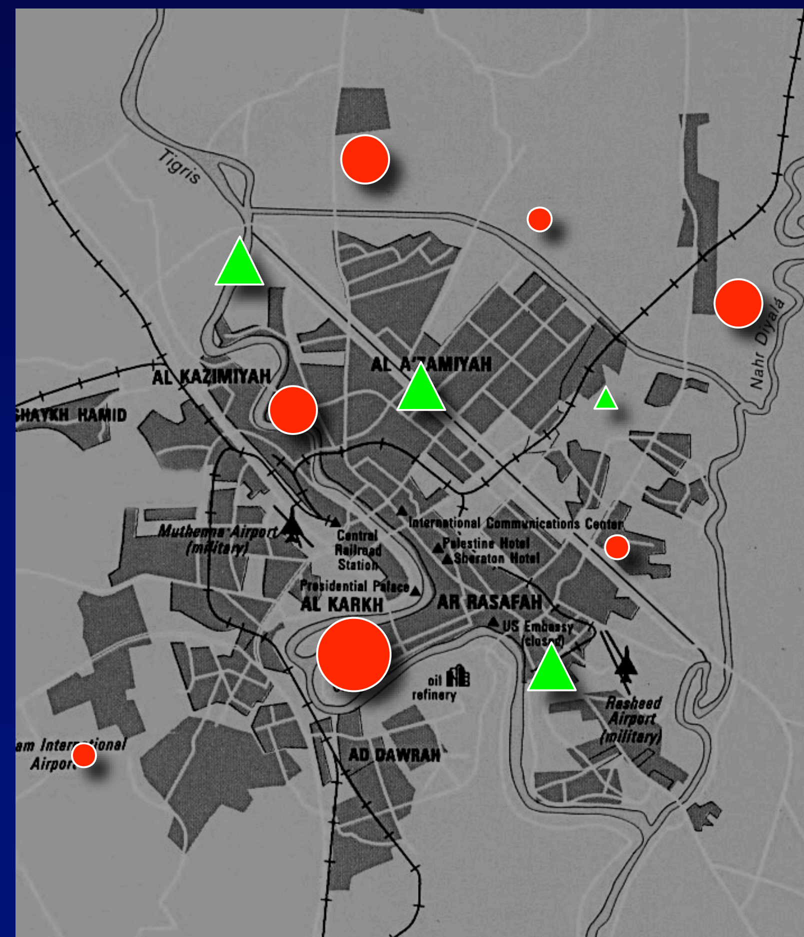
Periodic Surveillance

- Given: many targets to monitor with one/few aircraft
- Objective: early change awareness
- Decide: where to go next

Example: fire detection



Visit least stable “neediest” targets more often or never



Surveillance task performance

A decision-theoretic approach

$$ECI = \sum_{T}^{\text{Targets}} \sum_{i}^{\text{Intervals}} \int_{t=t1}^{t2} p(t) \cdot Cost(t2 - t)\,dt$$

ECI: expected cost of ignorance
Targets: locations to be monitored for some event
Intervals: period between successive observations
p(t): probability density function for event
Cost(t): expected cost if event occurs at time t

Goal of surveillance planning is to minimize ECI

Measuring Surveillance Performance

Fire example

Probability of occurrence (pdf)

$$p(t) = ae^{-at}$$

exponential

sigmoid

Cost of occurrence

$$\mathrm{cost}(d) = c_0 + \left(\frac{2}{1+e^{-k(d+l_1+l_2)}} - 1\right)(m - c_0)$$

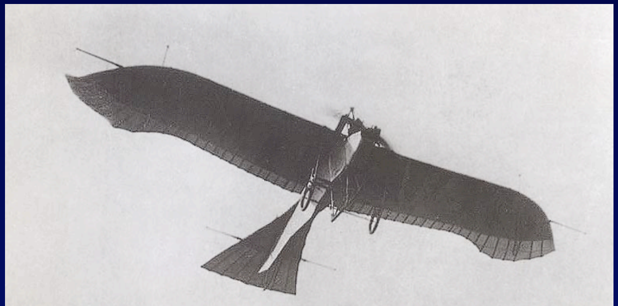
Expected cost of ignorance [t_1 t_2]

$$\mathrm{ECI}_{\tau}\ (\mathrm{t1},\ \mathrm{t2},\ \mathrm{a},\ \mathrm{k},\ \mathrm{m}) = \int_{t=t1}^{t2} ae^{-at} m\left(\frac{2}{1+e^{-k(t2-t)}} - 1\right)dt$$

Mission-level Autonomy Focus

Periodic Surveillance

- State of practice:
  Remotely piloted UAVs
- Why autonomy?
  - Fatigue: long/uneventful tasks hard on human operators
  - Effectiveness: people poor at complex optimization problems
  - Integration: autonomy + better, cheaper UAVs can function as part of unsupervised sensor net


Rumpler Taube, 1914

Autonomous Surveillance

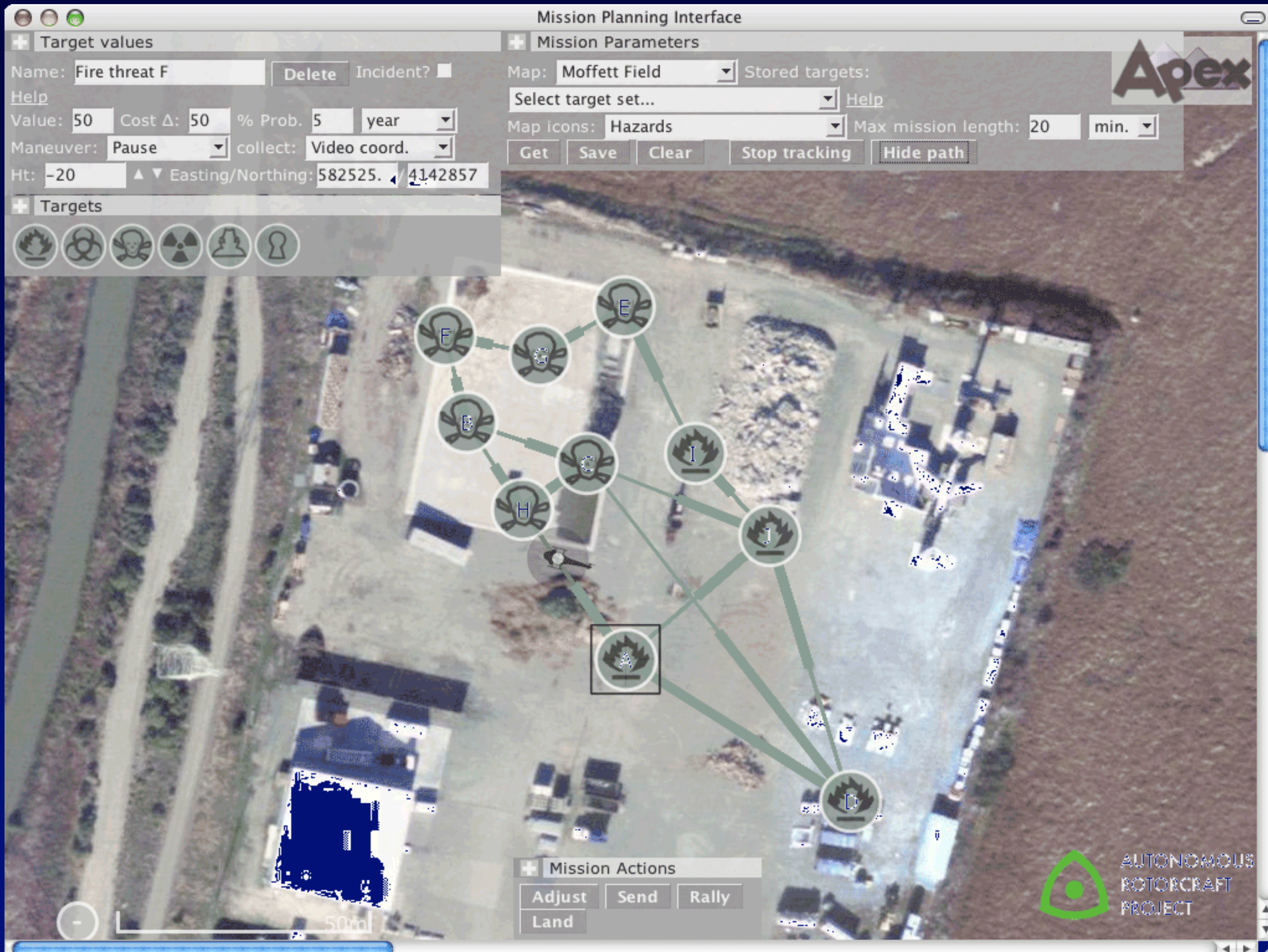
Objective: fully autonomous optimal surveillance

1. Generating mission plans
2. Executing mission plans in dynamic, uncertain conditions

Outline

- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Next steps

Specifying mission goals



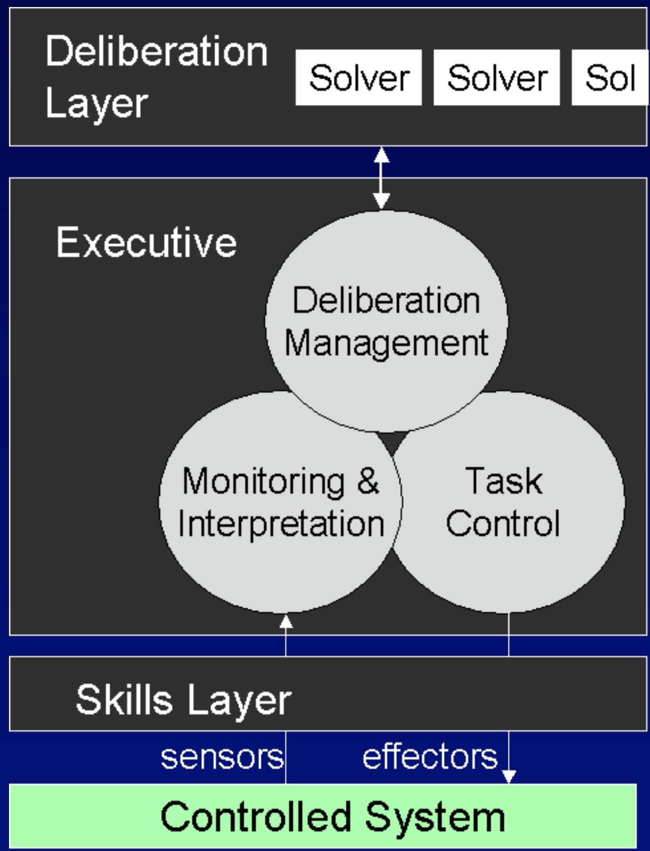
Autonomy Architecture

E.g. Autonomous Rotorcraft Project

Deliberation Layer
Periodic surveillance planning

Goal Executive Layer
Basic plan execution
Tactical observation maneuvers
Monitoring and anomaly-handling
Human interaction management

Skills (application-specific)
Autopilot
Payload controllers



Mission Planning challenges

1. Creating effective surveillance planning algorithm(s)
2. Determining at runtime which planning algorithm to use
3. Metrics: how well are we doing compared to state of practice (human-directed surveillance)?

Surveillance Planning Algorithms

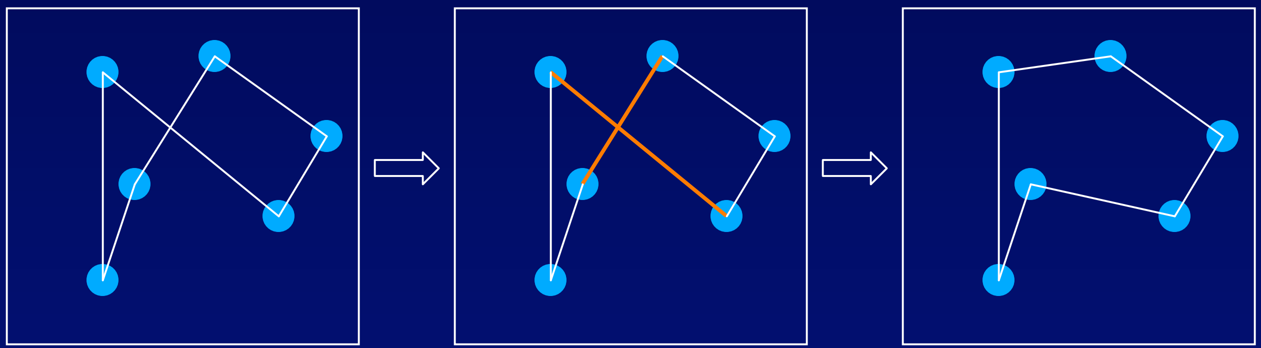
- Traveling Salesman Problem (TSP)
- Orienteering Problem
  - Time maximum (visit only subset of targets)
  - Reward varies for individual targets
- Surveillance Problem
  - Repeat visits yield multiple rewards
  - Reward value time-varying
  - Traverse time-cost state-dependent

Planning Approach #1

Best Cycle (local search)

Modified 2-OPT Exchange algorithm

- Basic 2-OPT computes approximate solutions for TSP
- Approach: start with a random tour; iteratively find and apply a tour-improving exchange of 2 tour segments until none found

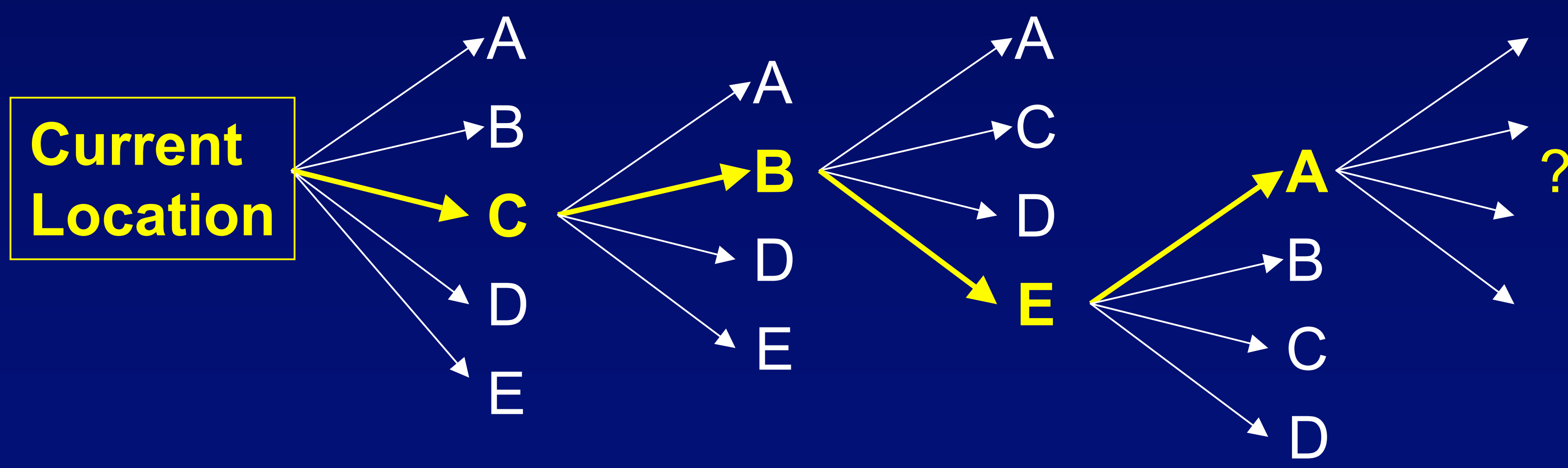


- Modifications
  - Use UAV kinematics model ("smoother") to compute traverse time
  - Evaluate return-to-home point given maximum flight duration = 60 minutes

Planning Approach #2

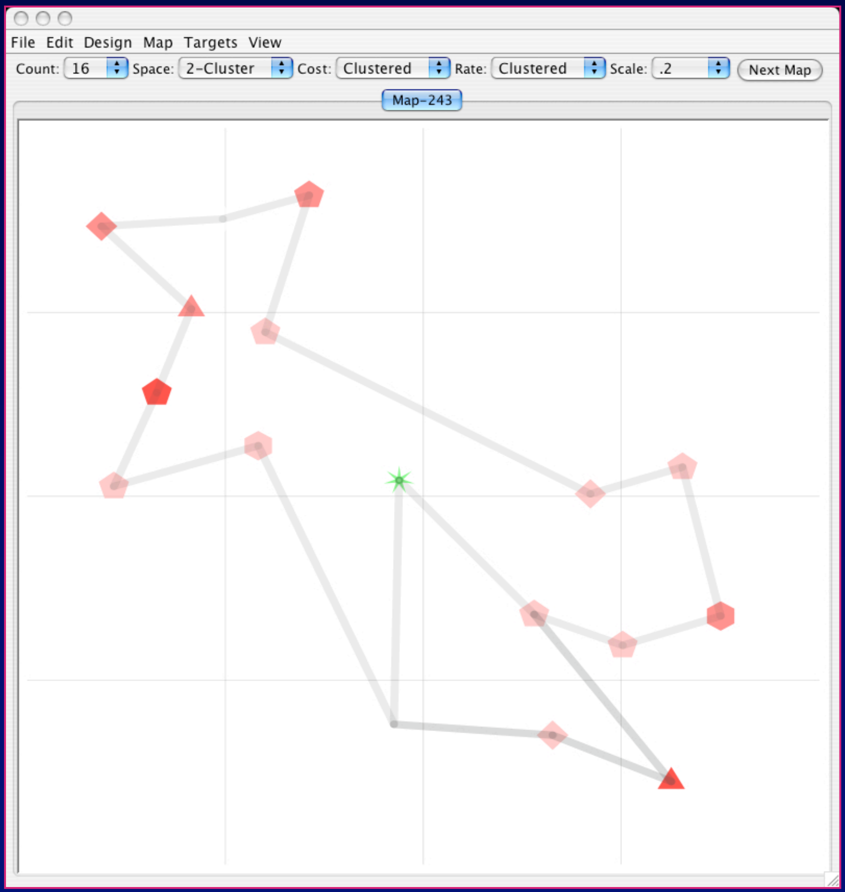
Best Path (heuristic depth-first search)

- Best-first (WAM), beam-search (IMMP)
- Heuristic transit time (accurate), visit-recency, cost-max



Planning Approach #3

Human as planner (state of practice)



UI for human subjects in surveillance performance experiment

Experiment Design

<u>243 Conditions, 5 IVs</u>

1. Number of targets
2. Spatial Distribution
3. Spatial Scale
4. Maxcost Distribution
5. Cost-Rate Distribution

- 7 subjects
- One trial per condition
- Randomly ordered
- Given training, practice and scoring decision aid
- ~6 hours / subject

Picking the Best Surveillance Planner

Min of Best			Count Space								
			4			8			16		
Scale	Rate	Cost	2-Cluster	Globular	Uniform	2-Cluster	Globular	Uniform	2-Cluster	Globular	Uniform
Large	Clustered	Clustered	1	1	0	3	3	1	1	3	1
		Fixed	1	1	0	3	3	1	2	3	2
		Uniform	1	2	1	1	2	1	1	1	1
	Fixed	Clustered	1	1	0	1	1	1	1	3	1
		Fixed	1	1	0	3	1	1	2	3	2
		Uniform	1	1	1	1	2	1	1	3	1
	Uniform	Clustered	1	1	2	1	3	1	1	3	1
		Fixed	1	1	2	3	3	1	2	3	1
		Uniform	1	1	1	1	2	1	1	3	1
Medium	Clustered	Clustered	2	2	0	1	2	2	2	2	2
		Fixed	1	2	0	2	2	0	2	2	2
		Uniform	1	2	2	2	2	2	2	2	2
	Fixed	Clustered	1	2	0	2	2	0	2	2	2
		Fixed	1	2	0	2	2	0	2	2	2
		Uniform	1	2	2	2	2	2	2	2	2
	Uniform	Clustered	1	2	1	2	2	2	2	2	2
		Fixed	1	2	2	2	2	2	2	2	2
		Uniform	1	2	0	2	2	2	2	2	2
Small	Clustered	Clustered	2	1	0	2	0	0	2	2	2
		Fixed	2	1	0	2	1	0	2	2	2
		Uniform	2	2	3	2	2	2	2	2	2
	Fixed	Clustered	2	1	0	1	2	2	2	2	2
		Fixed	2	1	0	2	2	2	2	2	2
		Uniform	2	1	0	2	1	0	2	2	2
	Uniform	Clustered	2	2	0	1	2	2	2	2	2
		Fixed	2	2	0	1	2	2	2	2	2
		Uniform	1	1	0	2	2	2	2	2	2

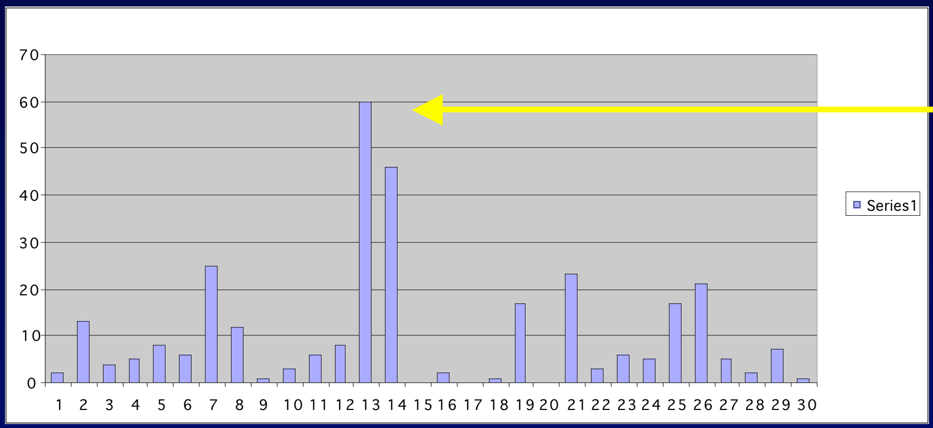
Human
2-Opt
WAM
no diff

Human Performance

- Algorithms significantly out-performed humans overall (4.9% vs. 2-Opt; $p < 0.01$)
- Human subjects differed significantly ($p < 0.05$)
- Humans did especially poorly with small-scale maps, small N, low spatial structure (uniform distribution)
- Human subjects made large errors in a small
- percentage of cases

Results reinforce value of autonomous surveillance planning

Magnitude of differences in algorithm performance



% difference between WAM, 2-OPT

Max Performance Difference = 60%

2-Opt: small, 5, perimeter

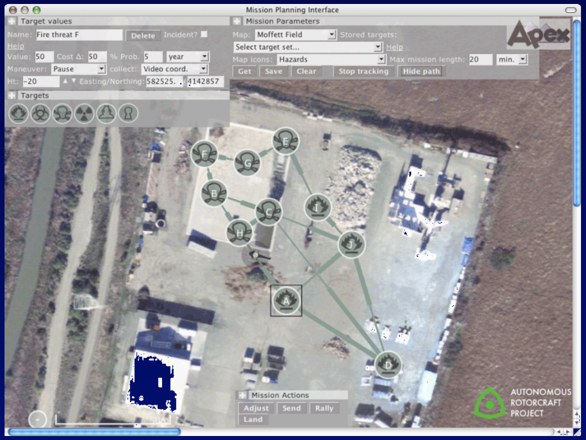
Min Performance Difference = 0%

<u>30 conditions, 100 trials per condition</u>

1. Number of targets: 5, 10 , 20
2. Geometry: uniform, globular, perimeter, 2-cluster, 3-cluster
3. Mission space: small, large

Automatic algorithm selection

Goal: select best algorithm given MPI-defined mission



Method:

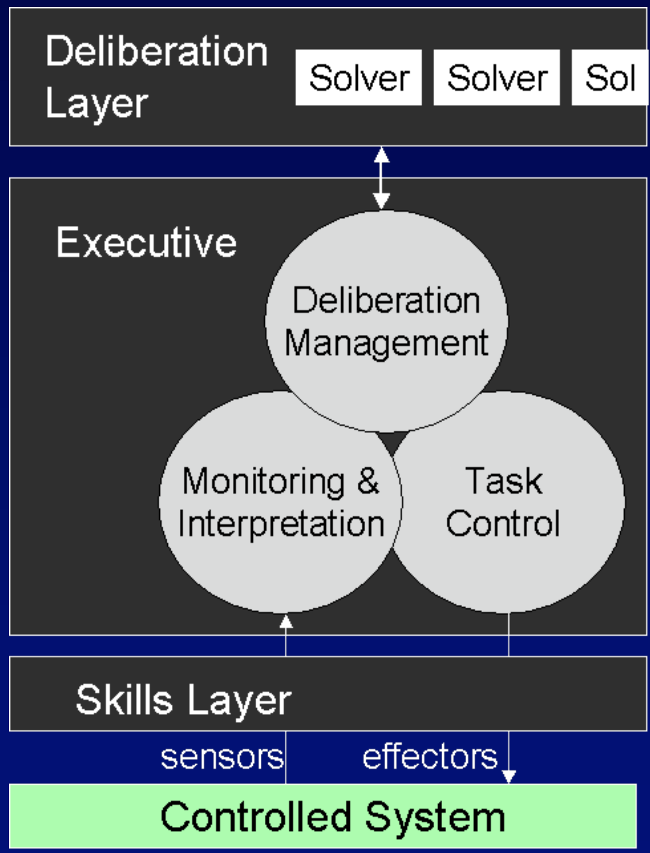
- Define standard mission classes
- Pre-compute performance of all available algorithms for all mission types; create preference table

3. Rapidly classify current mission at runtime (rapidly)
4. Index into preference table

Min of Best			Count Space								
			4			8			16		
Scale	Rate	Cost	2-Cluster	Globular	Uniform	2-Cluster	Globular	Uniform	2-Cluster	Globular	Uniform
Large	Clustered	Clustered	1	1	0	3	3	1	1	3	1
		Fixed	1	1	0	3	3	1	2	3	2
		Uniform	1	2	1	1	2	1	1	1	1
	Fixed	Clustered	1	1	0	1	1	1	1	3	1
		Fixed	1	1	0	3	1	1	2	3	2
		Uniform	1	1	1	1	2	1	1	3	1
	Uniform	Clustered	1	1	2	1	3	1	1	3	1
		Fixed	1	1	2	3	3	1	2	3	1
		Uniform	1	1	1	1	2	1	1	3	1
Medium	Clustered	Clustered	2	2	0	1	2	2	2	2	2
		Fixed	1	2	0	2	2	0	2	2	2
		Uniform	1	2	2	2	2	2	2	2	2
	Fixed	Clustered	1	2	0	2	2	0	2	2	2
		Fixed	1	2	0	2	2	0	2	2	2
		Uniform	1	2	2	2	2	2	2	2	2
	Uniform	Clustered	1	2	1	2	2	2	2	2	2
		Fixed	1	2	2	2	2	2	2	2	2
		Uniform	1	2	0	2	2	2	2	2	2
Small	Clustered	Clustered	2	1	0	2	0	0	2	2	2
		Fixed	2	1	0	2	1	0	2	2	2
		Uniform	2	2	3	2	2	2	2	2	2
	Fixed	Clustered	2	1	0	1	2	2	2	2	2
		Fixed	2	1	0	2	2	2	2	2	2
		Uniform	2	1	0	2	1	0	2	2	2
	Uniform	Clustered	2	2	0	1	2	2	2	2	2
		Fixed	2	2	0	1	2	2	2	2	2
		Uniform	1	1	0	2	2	2	2	2	2

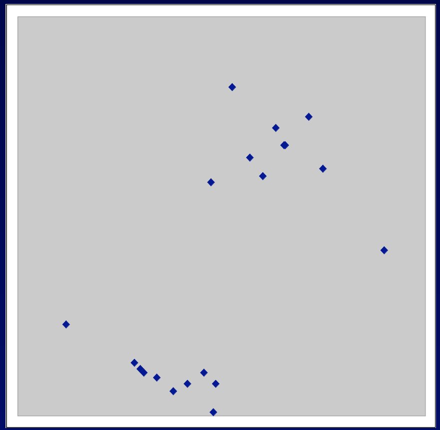
Reasoning and Control Services

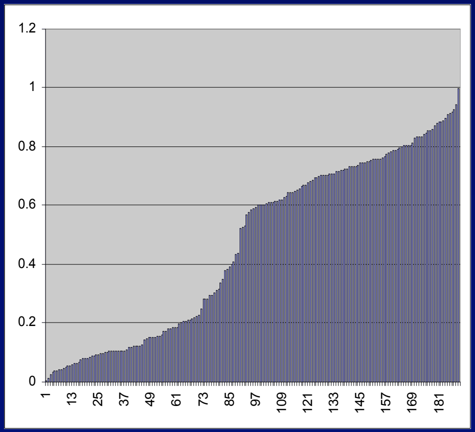
Dispatch
Signal and process handling
Condition detection
Memory management
Refinement
Transformation
Projection
Self-callibration
Deliberation control



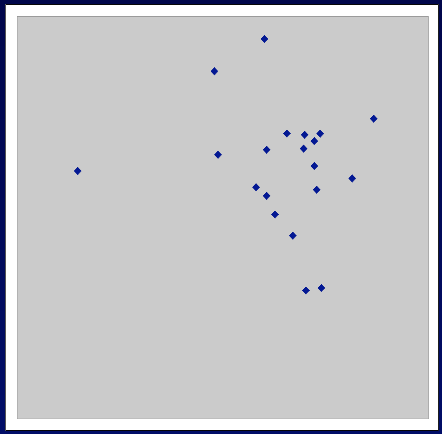
Classifying Target Set Geometries

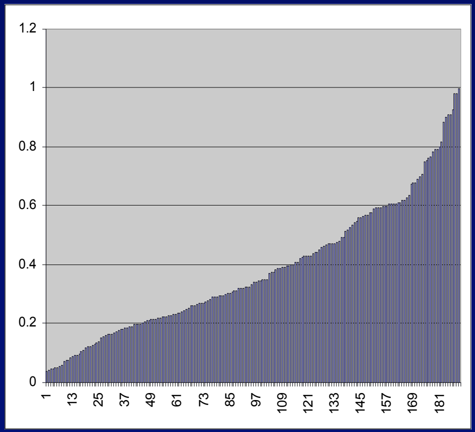
Multicluster



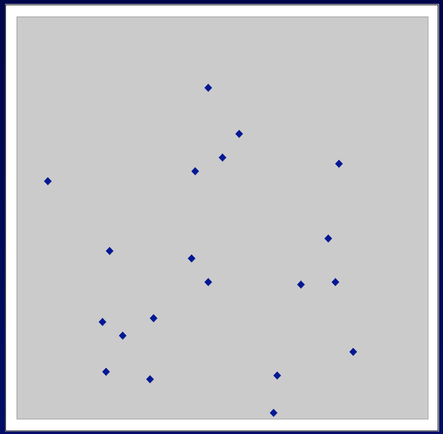


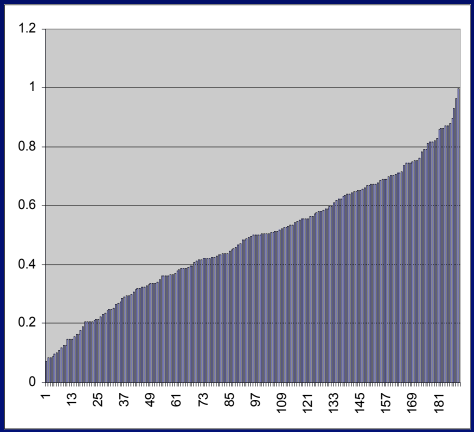
Globular





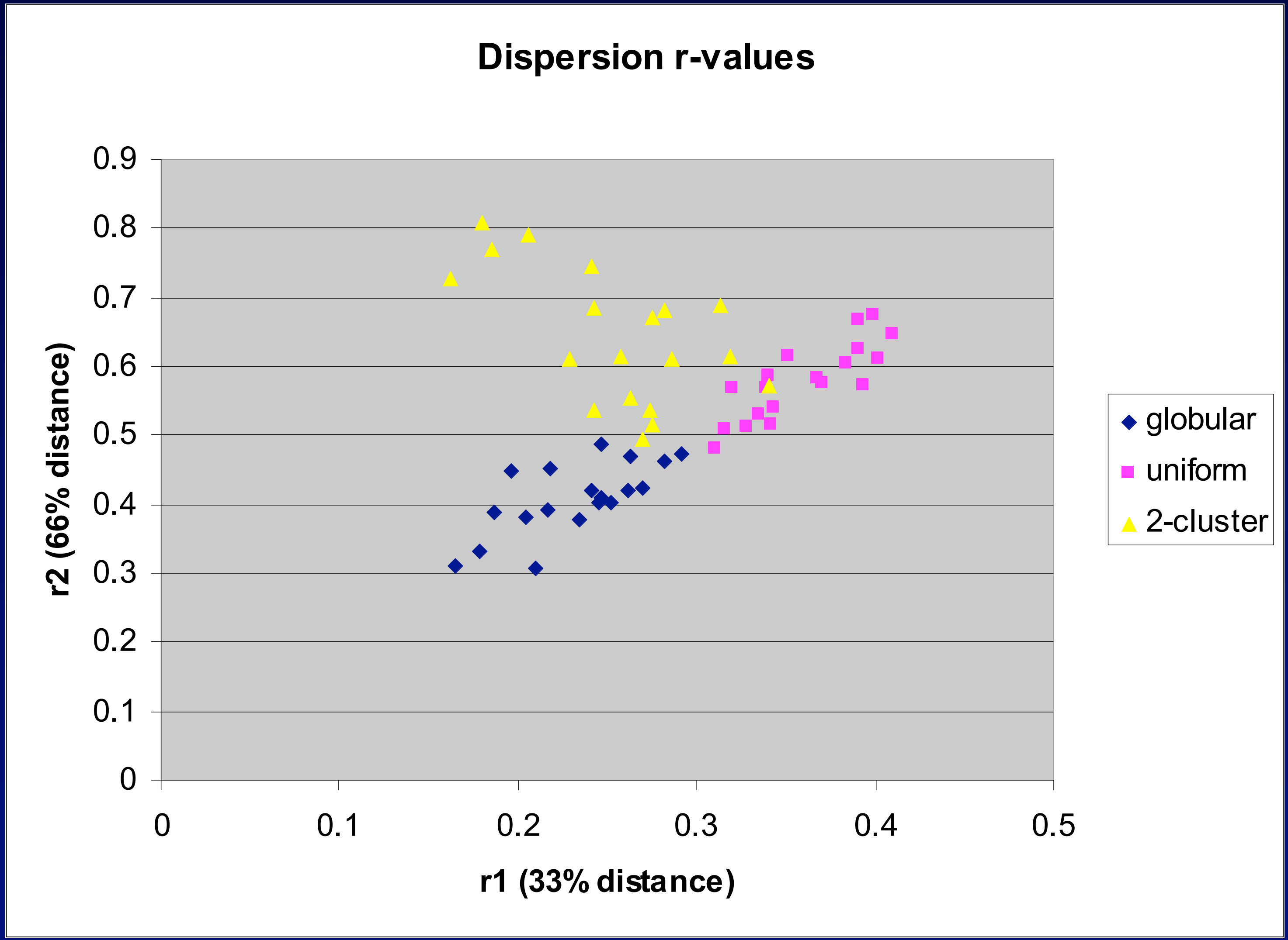
Uniform



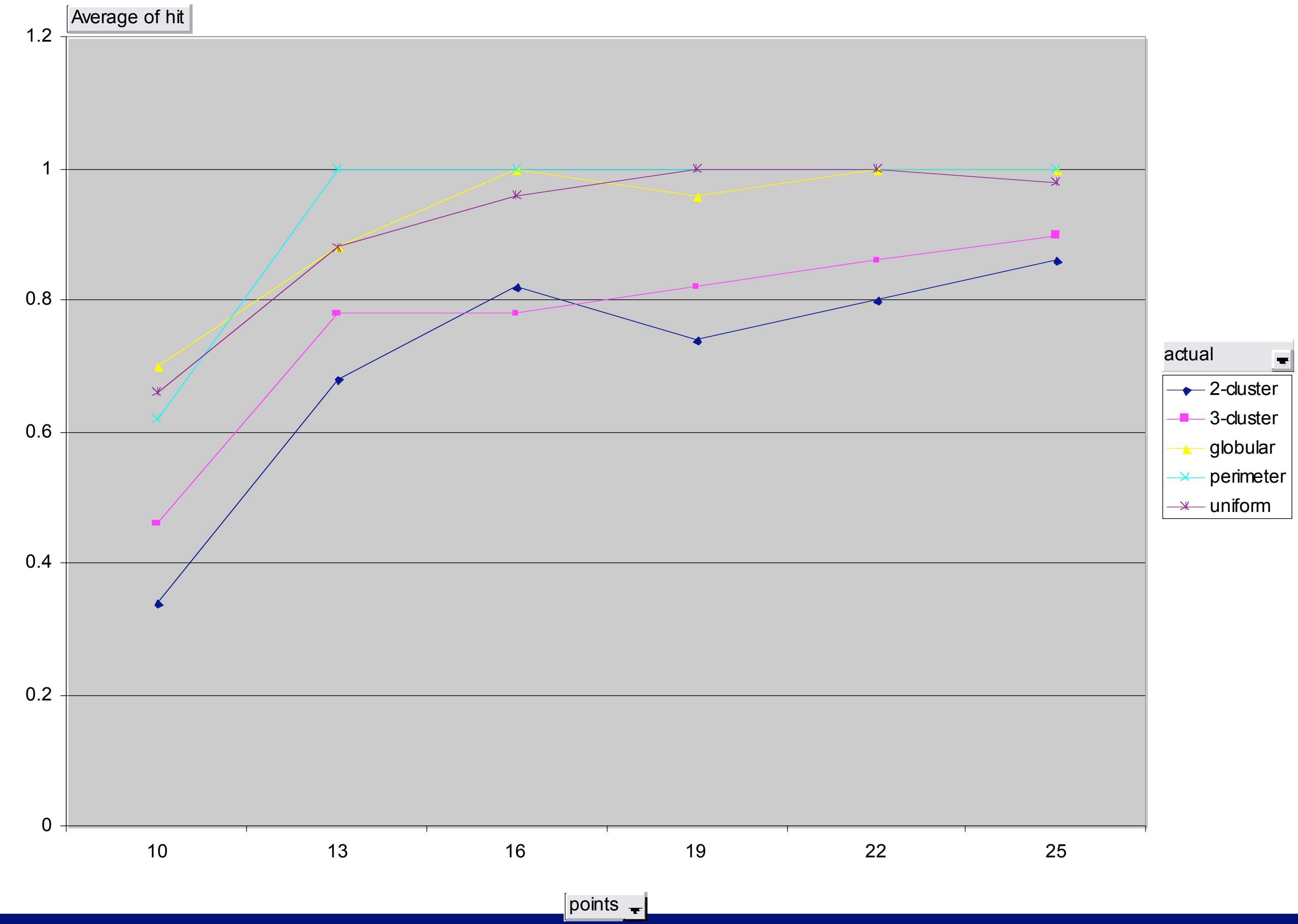


"Distograms" of sorted, normalized pair distances

Classifying Dispersion Patterns



Classifier Results



Outline

- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Visualizing autonomy logic

Observing Targets

Goal

acquire *sensor data products* in support of surveillance task.

Observation behavior elements

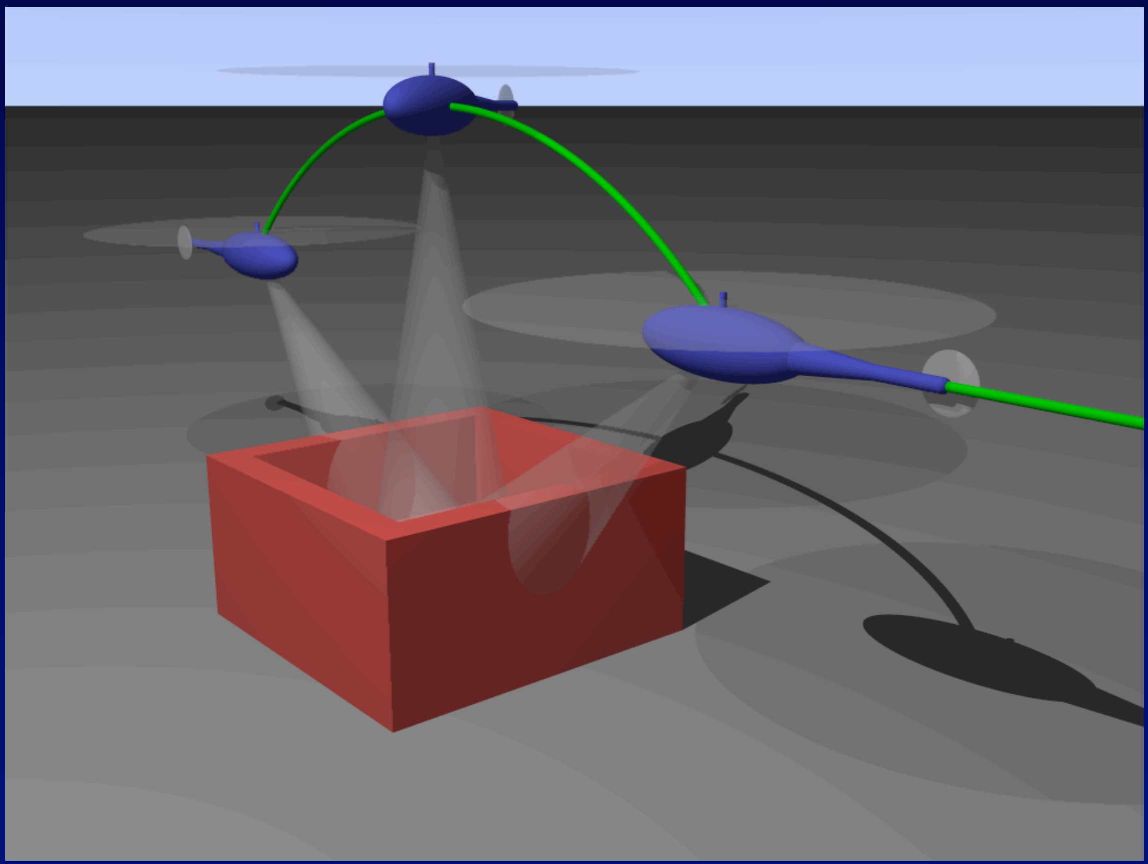
- Path
- Attitude mode changes
- Speed changes
- Sensor payload behavior (powerup, actuation, imaging)
- Data handling behavior (storage, compression, telemetry)

Challenges

1. Large space of observing behaviors and need to link behavior to surveillance needs requires AI planning
2. Too much uncertainty for detailed advanced planning

Observation Behaviors

Arch with camera tracking



Compensating for limited camera actuation range

Sequence

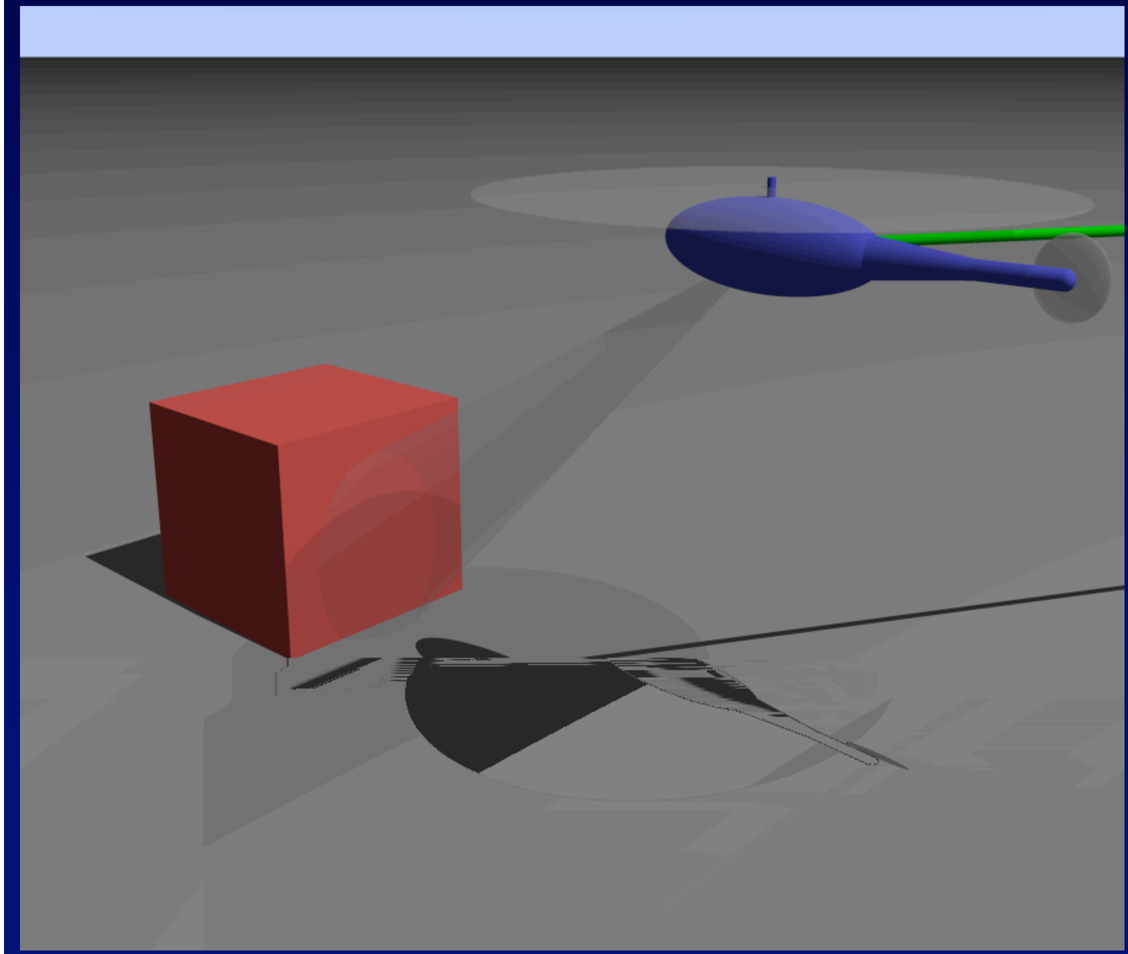
- move to standoff
- turn on video camera and track target
- climb to apex
- reduce speed
- reverse heading
- increase speed
- descend arch
- turn off camera

Parameters

- View radius

Observation Behaviors

Pause and Stare (Best Vantage)

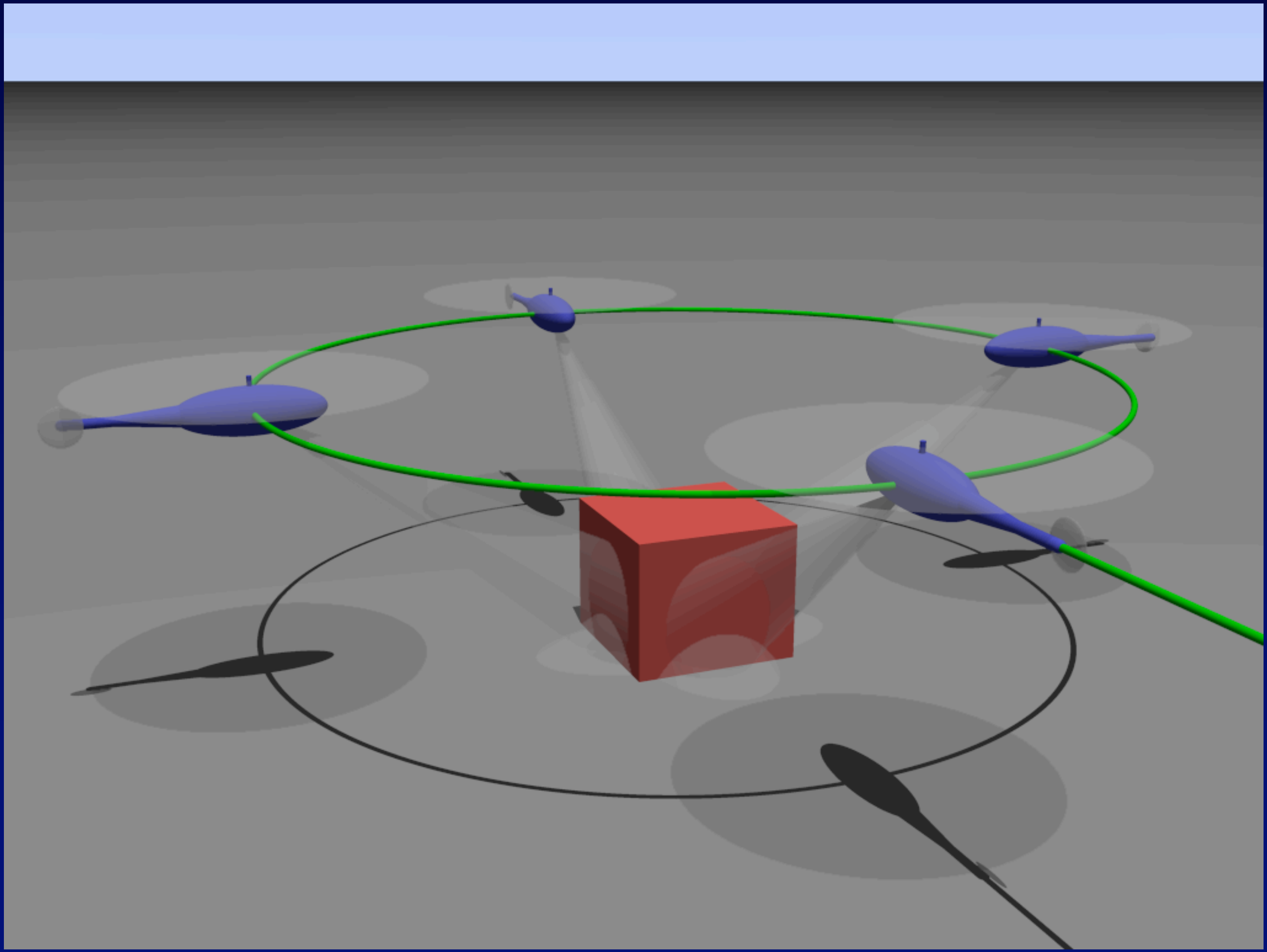


View position factors
- Current sun position (shadows in image)
- Camera resolution
- Wind speed/direction
- Obstacles (line of sight)

Parameters
- Pause duration
- Sensing action (image, video, laser sweep)

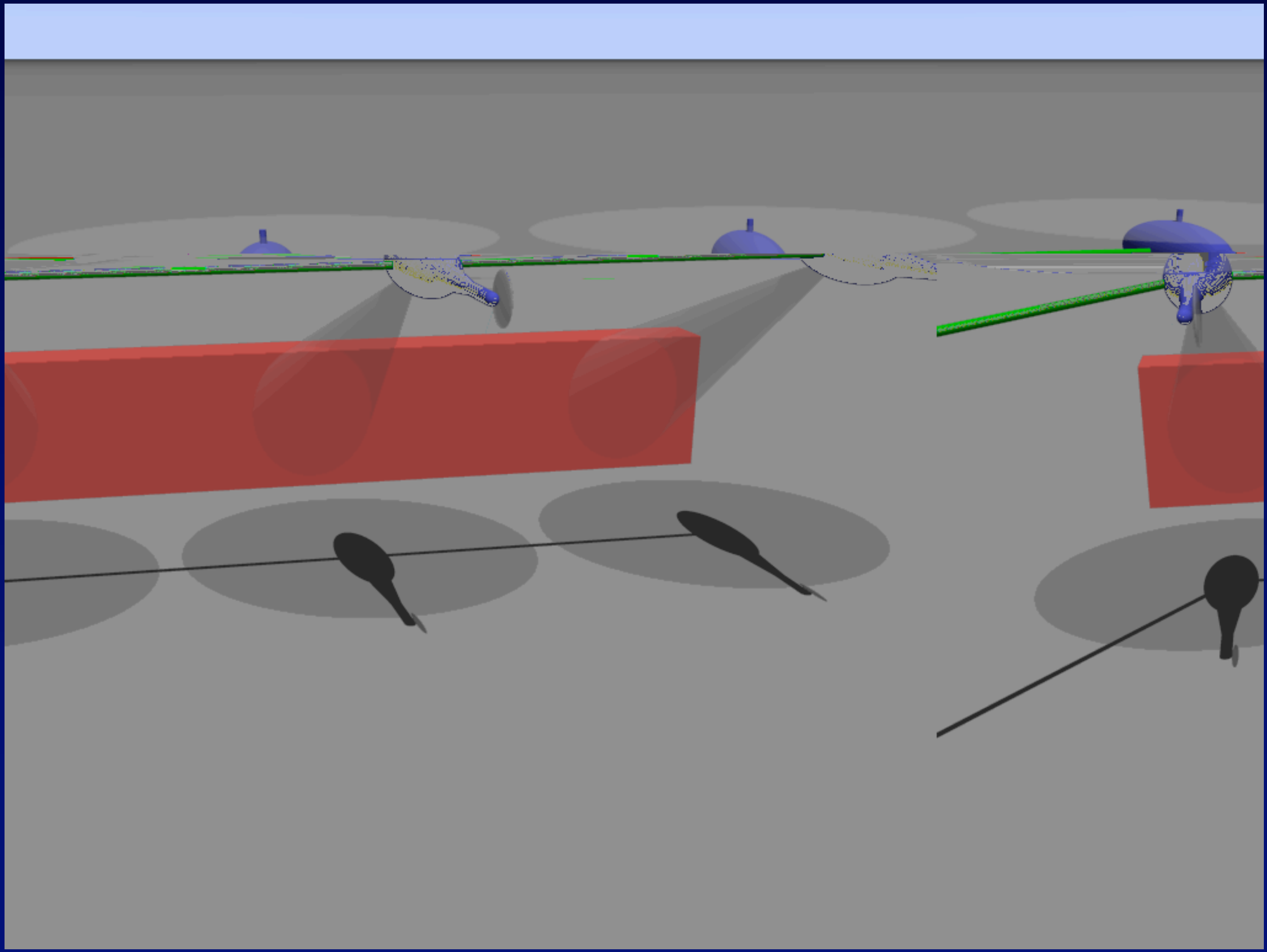
Observation Behaviors

Pirouette



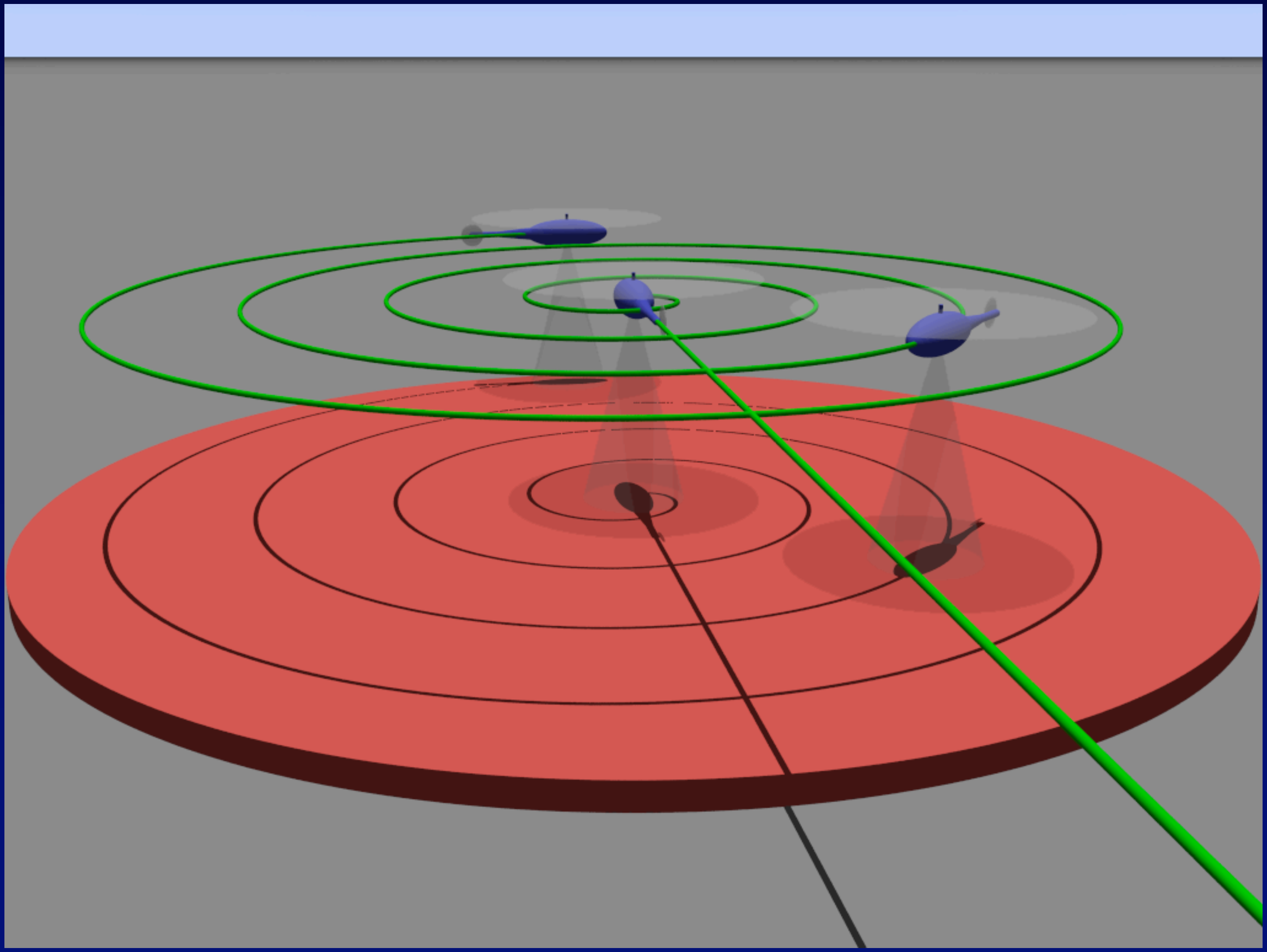
Observation Behaviors

Scan



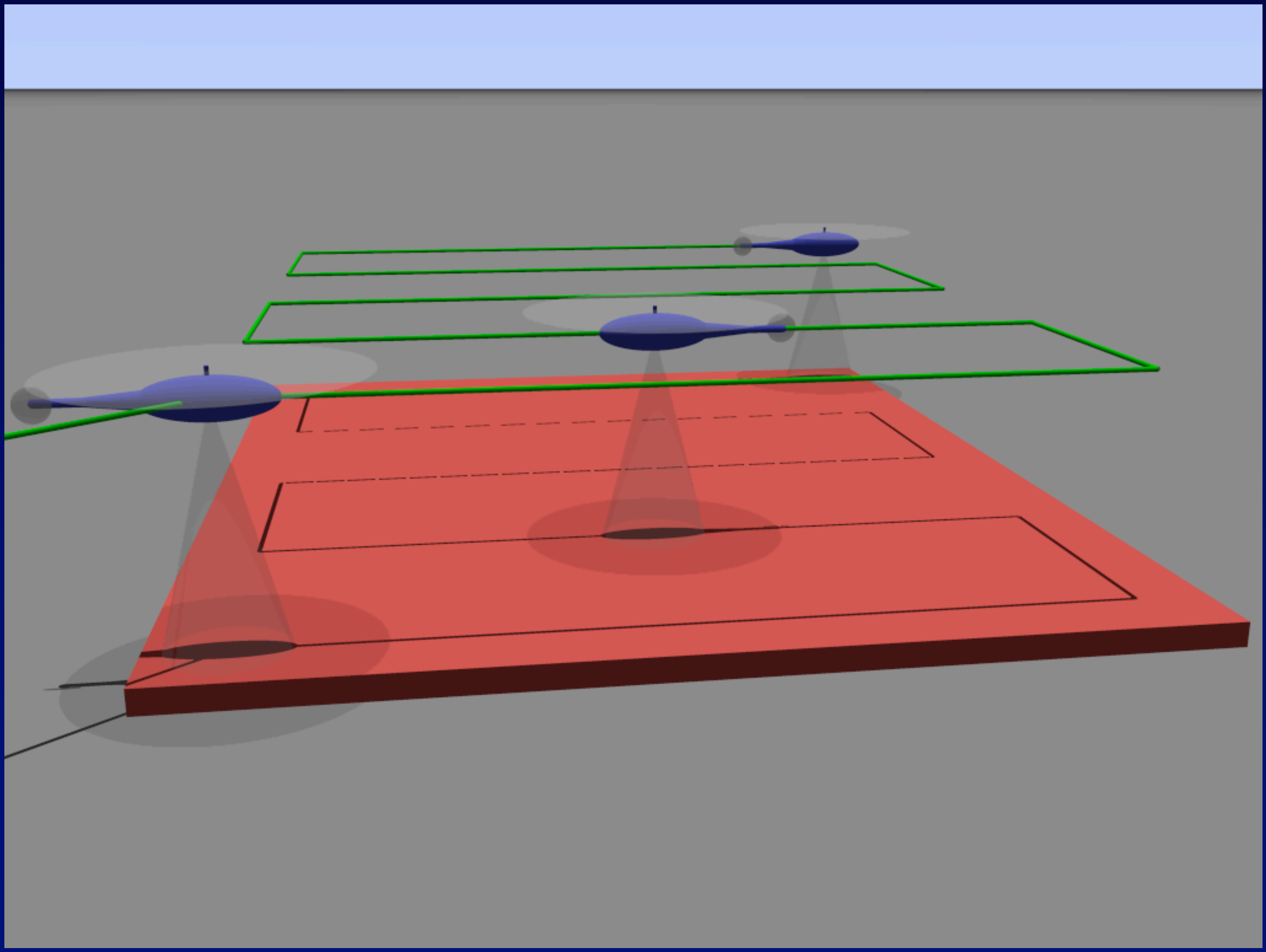
Observation Behaviors

Spiral



Observation Behaviors

Area Sweep



Observation Behaviors
Challenge

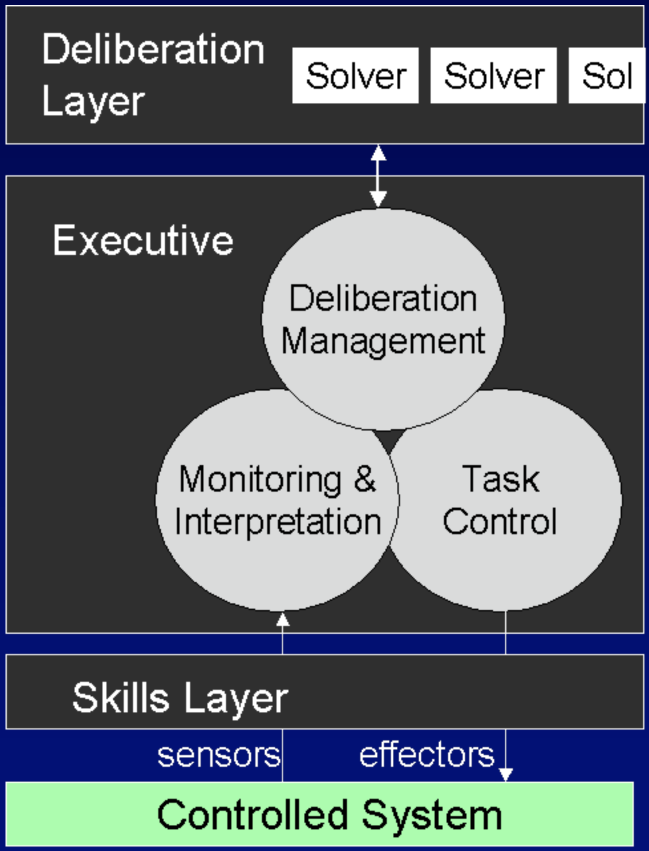
Observation behaviors can't always be planned out in detail in advance due to uncertainty about:

- Sun position (time of day)
- Wind
- View and path obstacles
- Most useful data product

It is useful to be able to leave the exact behavior unspecified in the mission plan until it is almost time to observe.

Reasoning and Control Services

Dispatch
Signal and process handling
Condition detection
Memory management
Refinement
Transformation
Projection
Self-callibration
Deliberation control



Outline

- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Visualizing autonomy logic

Adaptation to unplanned conditions

Planning is everything. Plans are nothing. (D.D. Eisenhower)

Problem: almost any condition not precisely anticipated can invalidate a plan or reduce its effectiveness.

- Operator interventions
  - Interval of manually controlled observation*
  - Change to mission goals or parameters*
- Unexpected outcomes
  - Quality of data product at target less than desired
  - Unexpectedly long/short time to traverse; fuel consumed
- System and operational environment contingencies
  - Loss of communication signal strength*
  - Loss of camera power
  - Shift in wind

* Illustrated in flight test

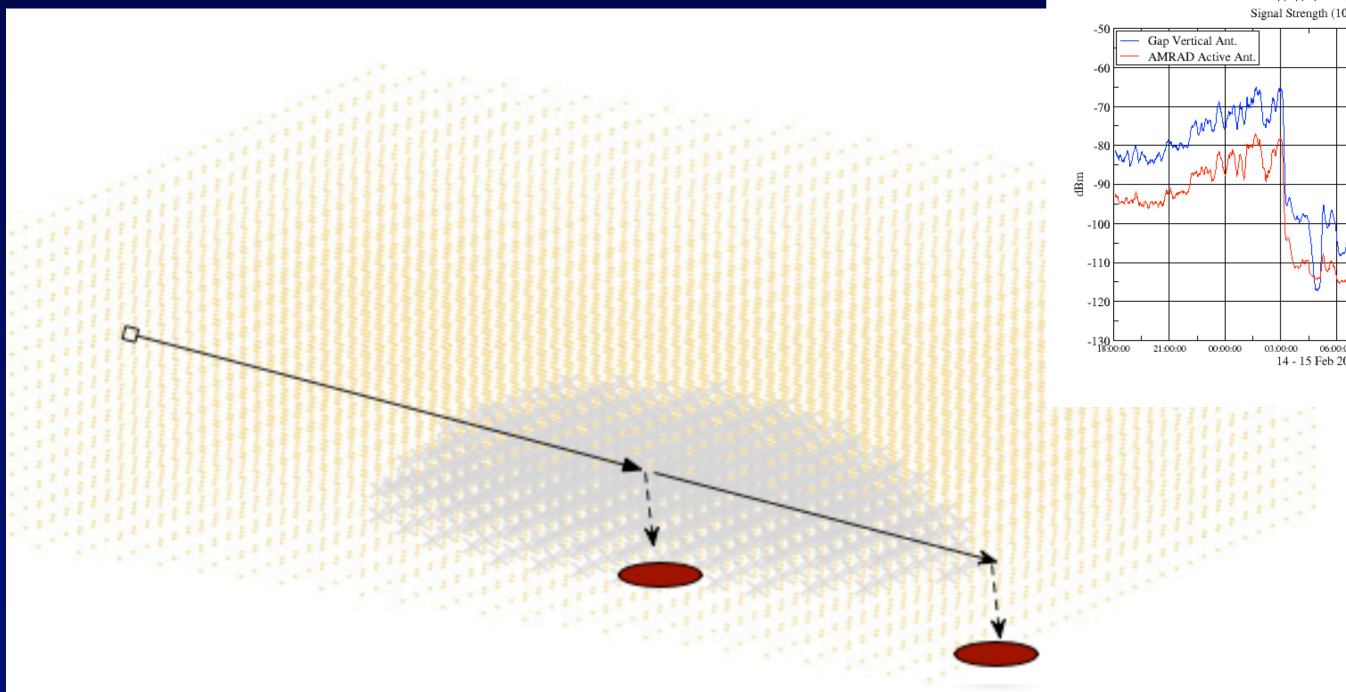
Adaptation to unplanned conditions

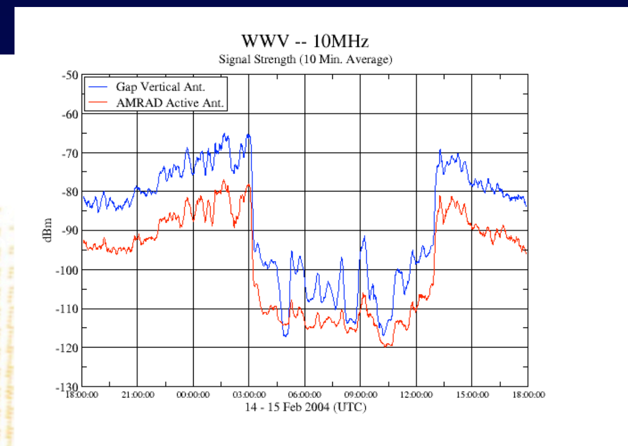
Handling anomalies

1. Detect the condition
2. Recover (if needed)
3. Determine modification to current plan to cope with the condition (if needed)
4. Assess impact of recovery/modification, then either:
   - Ignore anomaly and continue
   - Modify plan and continue
   - Throw out old plan and generate a new one

Adaptation to unplanned conditions

Example: Loss of communication signal strength

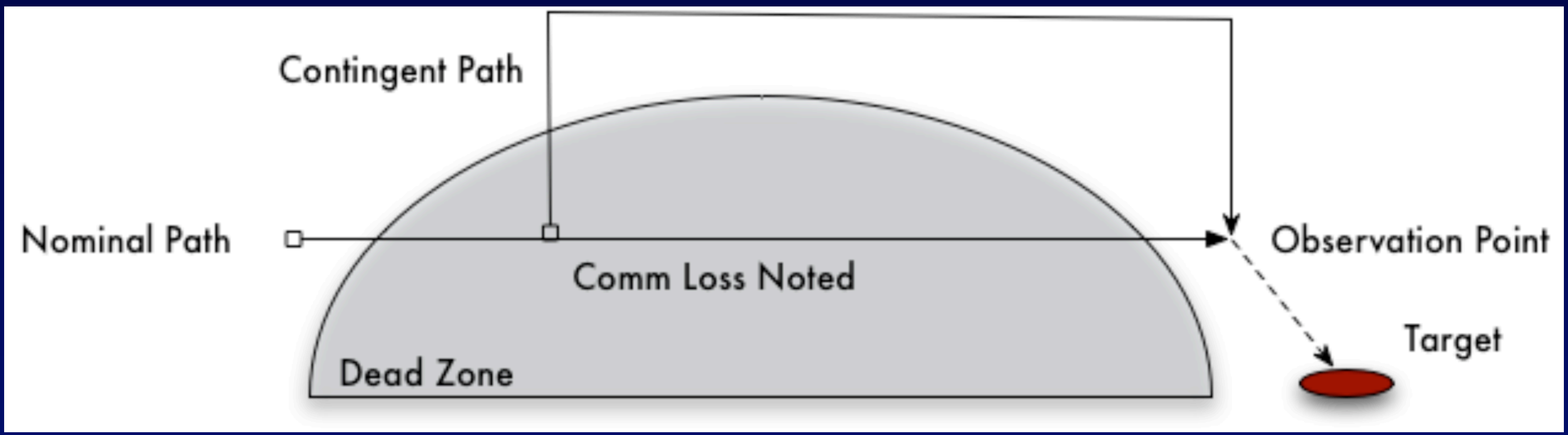




Step 1: In transit to next target, loss of signal strength of sufficient magnitude and duration to trigger operational contingency occurs.

Adaptation to unplanned conditions

Example: Loss of communication signal strength

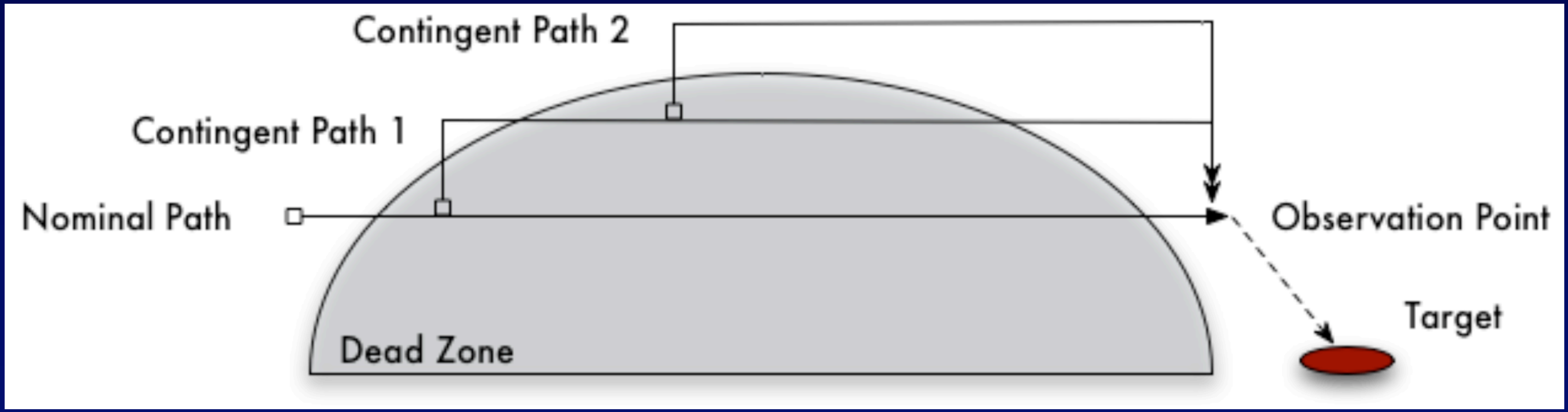


Step 2: Apex modifies plan: climb above comm loss area (and above preferred observation altitude); transit to next target; descend.

Step 3: Apex assesses impact of modification on mission plan. If time cost reduces number of targets reachable before mission end or violates plan constraint, then replan.

Adaptation to unplanned conditions

Example: Loss of comm signal strength

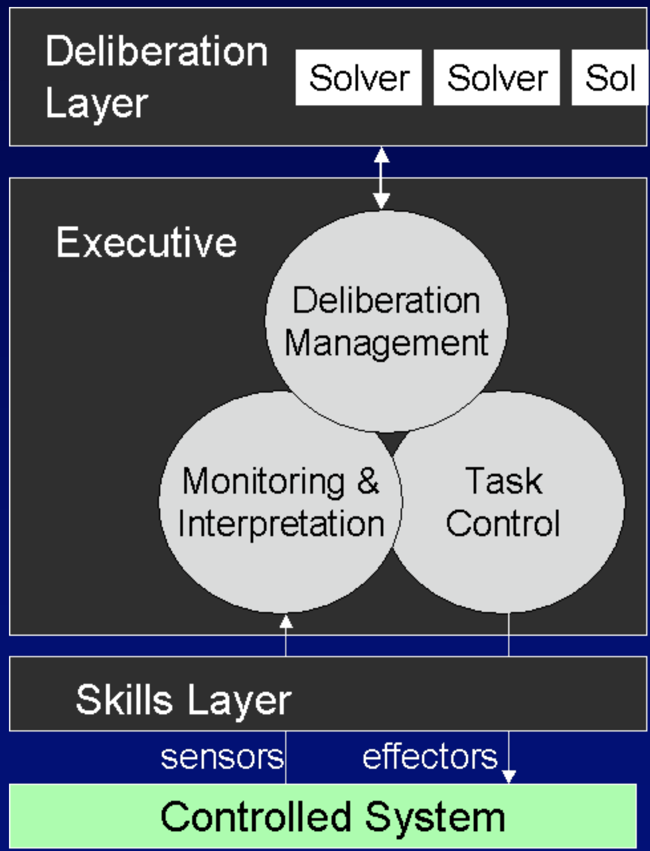


If comm loss area extends higher than expected, additional climbing step(s) may be inserted into plan (requiring new replan assessment)

Adaptation to unplanned conditions

Reasoning and Control Services

Dispatch
Signal and process handling
Condition detection
Memory management
Refinement
Transformation
Projection
Self-callibration
Deliberation control



Outline

- Apex: autonomy software overview
- Autonomous surveillance missions
- Automatically generating mission plans
- Observing targets
- Adapting to unplanned conditions
- Visualizing autonomy logic

Apex Usability

Indirect contributions to the capabilities we've developed for ARP

- Behavior representation language (PDL)
- Debugging / critiquing (Sherpa)
- Application configuration
- User support

Autonomy Visualization

- Visualizing behavior vs. visualizing logic
  - Behavior: what is happening, what did happen
  - Logic: what might happen or might have happened
    - Causal Explanation (complex behavior, incorrect behavior)
    - Predicting (planned, contingent futures)
- 4 points of view
  - Autonomy application developer
  - Systems engineer
  - Operator
  - Stakeholder

Sherpa

Integrated Debugging and Demonstration Environment

Browser interaction model for viewing data

8 ways to view autonomy logic

- Inspect
- Trace
- Diagram
- PERT (schedule)
- Agenda (tree)
- PDL (template)
- Monitor
- State Variable

forward/back buttons

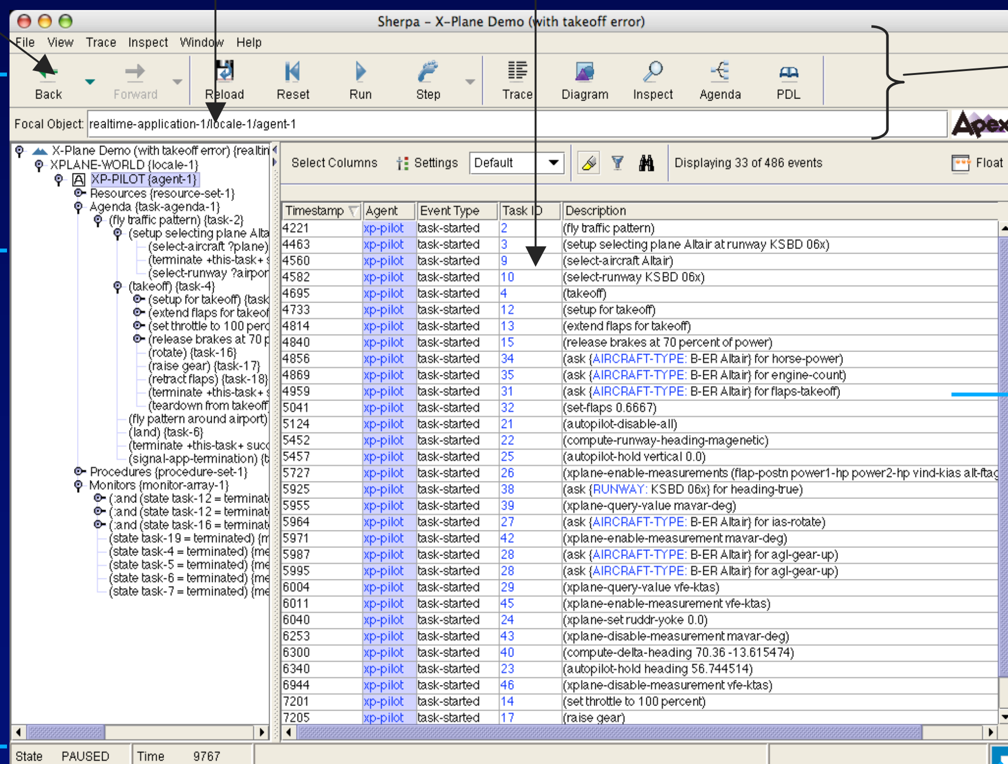
URL focal object

All data objects as hypertext

Main Toolbar

Object Tree (Navigation) Window

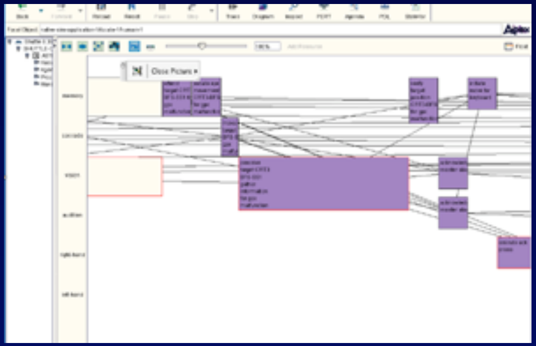
Application Status Bar



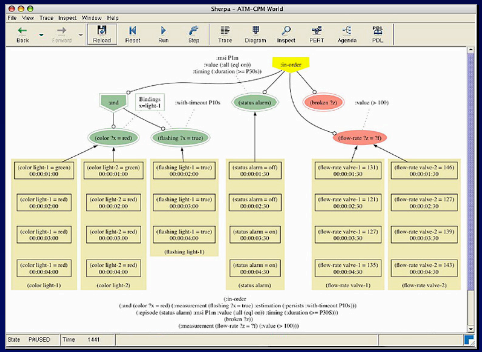
Main View Window

Communication Status Indicator

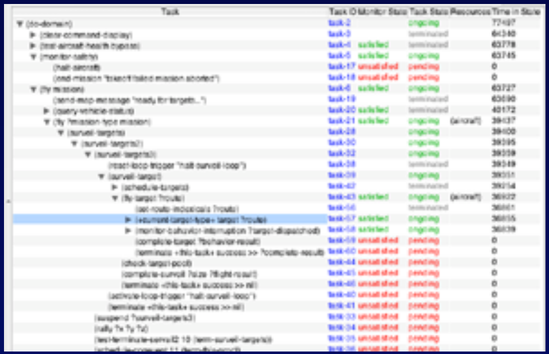
Sherpa views



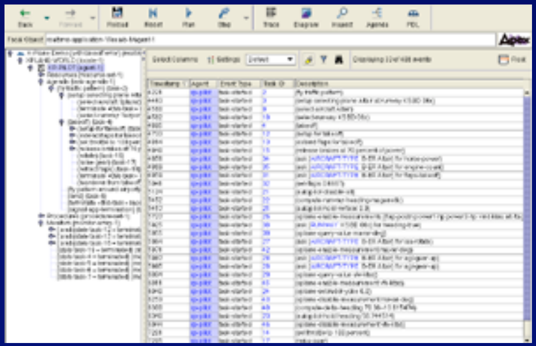
PERT chart



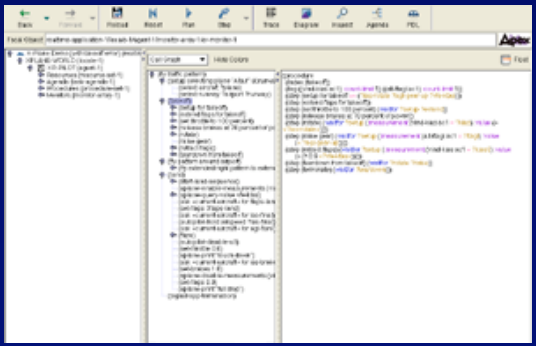
Event Monitoring
Logic / History



Task Agenda



Event Trace



Behavior Specifications

Conclusion

- Apex provides capability for full mission autonomy in complex missions
- Surveillance planning is something best done by autonomous systems
- Executing plans would be easy if the world were predictable, but it’s not.
- Reusability is important: building capable, reliable, usable autonomy software is too difficult to do repeatedly for every new platforms and missions.

For more information

Web site http://ti.arc.nasa.gov/projects/apex/

- ARP project description, online MPI demo
- Publications
- Download software (open source)

Email: Michael.A.Freed@nasa.gov

Apex Outer-Loop Control Block Diagram